**Core Components**

1. **Redux Data Layer**: Complete state management with Redux Toolkit, featuring slices for auth, files, chat, dashboard, tenants, and users.
2. **UI Framework**: Built on TailwindCSS with a comprehensive component library including buttons, cards, inputs, and specialized components.
3. **Authentication**: Full authentication flow with login, registration, and session management.
4. **File Analysis System**:
   * File upload with local and cloud storage options (Google Drive, OneDrive, Dropbox, SharePoint)
   * File tagging and categorization
   * Grid and list views for files management
5. **Chat Interface**:
   * Conversation management
   * Real-time messaging with the RAG service
   * File context selection for contextual answers
   * Source attribution for answers
6. **Dashboard**: Analytics dashboard with key metrics and visualizations

**File Structure**

* Modern Next.js app router structure
* Component-based architecture
* Clean organization of hooks, Redux slices, UI components, and pages

**Deployment Guides**

* Azure deployment guide for production hosting
* CodeSandbox setup guide for development and testing
* Backend requirements document for API implementation

**Notable Features**

* Dark/light mode support
* Multi-tenant capabilities
* RBAC (role-based access control)
* Responsive design for all screen sizes
* Real-time user feedback on operations
* Detailed error handling

The implementation follows best practices for React and Next.js development, with TypeScript for type safety. The code is structured to be maintainable and extensible as the application grows.

To complete the project, you would need to implement the backend API according to the backend requirements document, focusing on LlamaIndex for RAG capabilities, FastAPI for the REST API, Weaviate for vector storage, and integration with your private LLM.